Amendments to the Specification

Please replace paragraph [0017] with the following amended paragraph:

[0017] Referring now to Figures 1, 2 and 3a, Figure 1 shows a preferred embodiment of a mobile phone 170 comprising an OMAP processor 172 coupled to a UART/USB port 174, a flash memory 178 and comprising a ROM code (i.e., on-chip firmware) 176. Figure 2 illustrates a flow diagram describing the process of binding a boot image to the mobile phone 170. Figure 3a illustrates a boot image 300 comprising a TOC field 302 that describes the contents of the boot image 300, a KEYS header 304 that comprises keys used for cryptographic reasons as described below and a common header 306 that acts as a header for a flash loader 308, which comprises a protected application ("PA") ~~318~~316. The boot image 300 also may comprise other fields 310, a PA 312 and an empty device-bound certificate ("DBC") field 314. Protected applications are thusly named because the protected applications operate in a secure-mode environment, which may be defined as a hardware-based secure execution environment that is generally tamper-proof.

Please replace paragraph [0019] with the following amended paragraph:

[0019] The flash loader 308 subsequently may load and call the PA ~~318~~316 (block 206). When calling the PA ~~318~~316, the flash loader 308 sends various parameters, comprising pointers to various components of the boot image 300 (e.g., the common header 306) as well as the values of Creator ID and Application ID found in the common header 306. The Creator ID describes the owner or creator of a DBC and the Application ID serves as an identifier for the application that creates the DBC. The PA ~~318~~316 may use these pointers and values as necessary.

Please replace paragraph [0020] with the following amended paragraph:

[0020] At least one purpose of the PA ~~318~~316 is to compute the DBC (block 208), optionally encrypt the DBC with a random key (block 210), and pass the DBC to the flash loader 308 for further processing (block 212). As previously discussed, the PA ~~318~~316

operates in a secure-mode environment. The PA ~~318~~ 316 may begin generating the DBC as follows:

$$HMAC = HMAC_{KEY}(SHA\text{-}1(\text{Common Header 306 + Boot Loader}) \,||\, \text{Public Chip ID} \,||\, \text{Creator ID} \,||\, \text{Application ID} \,||\, \text{Reserved Fields}),$$

where "HMAC" denotes a hashed message authorization code, the symbol "||" denotes concatenation, the Public Chip ID serves as a public identifier for the OMAP PROCESSOR 172, the Reserved Fields contain any information (e.g., an IMEI certificate) and the boot loader is contained in the boot image 300 as described in Figure 3b below. Specifically, the common header 306 and the boot loader are first hashed together using the commonly-known SHA-1 algorithm, described below. The result is concatenated with various data as shown above (e.g., Public Chip ID, Creator ID). The resulting concatenation is hashed using a key (i.e., KEY) by a commonly-known HMAC cryptographic algorithm, where KEY is generated as:

$$KEY = SHA\text{-}1(\text{Chip Specific ID} \,||\, \text{Creator ID} \,||\, \text{Application ID}),$$

and where the Chip Specific ID is a secret identifier created by the ROM code 176 or other system firmware and available only inside secure mode (i.e., during the execution of a PA). A secure hash algorithm SHA-1 is used for computing a "condensed representation" of a message or a data file. The "condensed representation" is of fixed length and is known as a "message digest" or "fingerprint." It is computationally infeasible to produce two messages having the same message digest. This uniqueness enables the message digest to act as a "fingerprint" of the message. For instance, SHA-1 may be used to ensure the integrity of a downloaded or received file by comparing the file hash with the original file hash. Any message or similar construct requiring integrity may be verified in this fashion.

Please replace paragraph [0021] with the following amended paragraph:

[0021] The PA ~~318~~ 316 completes the DBC computation by assembling a DBC as illustrated in Figure 3b using information computed by the PA ~~318~~ 316 or received from the flash loader 308. Specifically, the completed DBC may comprise a Public Chip ID 322, a Creator ID 324, an Application ID 326, a boot loader/common header hash 328, reserved fields 330 and an HMAC 332 generated as described above. The reserved fields 330 may be filled with an IMEI certificate 330 if an IMEI certificate was downloaded in block 204. The reserved fields 330 also may be filled with any other device-specific information. The PA ~~318~~ 316 then may optionally encrypt the DBC with a random, secret key K, computed as:

$$K = \text{SHA-1}(\text{Chip Specific ID} \,||\, \text{Creator ID} \,||\, \text{Application ID}).$$

Encrypting the DBC with a random, secret key K protects all of the contents of the DBC (e.g., the IMEI certificate 330). Once the DBC is encrypted or the encryption step is bypassed, the PA ~~318~~ 316 passes the DBC to the flash loader 308 for further processing.

Please replace paragraph [0022] with the following amended paragraph:

[0022] The flash loader 308 receives the DBC from the PA ~~318~~ 316 and inserts the DBC into the empty DBC field 314 (block 214), thereby establishing a DBC 314 inside the boot image 300. The flash loader 308 then completes the binding process by flashing (i.e., writing) the boot image 300 to the flash memory 178 of the mobile phone 170 (block 216).